Homework4

LinLeeCheong

March 5, 2016

## Problem #1

1. ------------X1<1-----------  
    |     |  
    |     |

 ---X2<1----     5  
 | |  
--X1<0--  15  
| |  
3 --x2<0---  
 | |  
 10 0



2

|  |  |  |
| --- | --- | --- |
| 2.49  1 | | |
| -1.06 | 0.21 | |
| -1.80  X2 | | 0.63 |

0

1

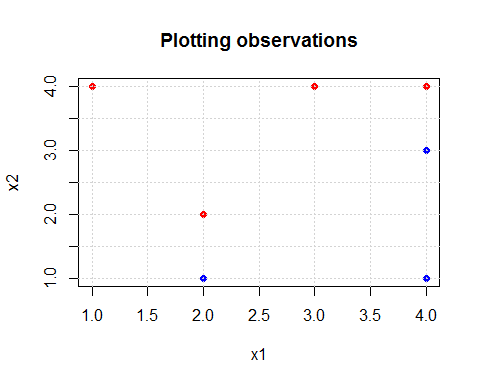
X1

## 

## Problem #2

1. Plot observations

x1<-c(3,2,4,1,2,4,4)  
x2<-c(4,2,4,4,1,3,1)  
y<-c('red','red','red','red','blue','blue','blue')  
plot(x1,x2,col=y,pch=19, main='Plotting observations')  
grid()

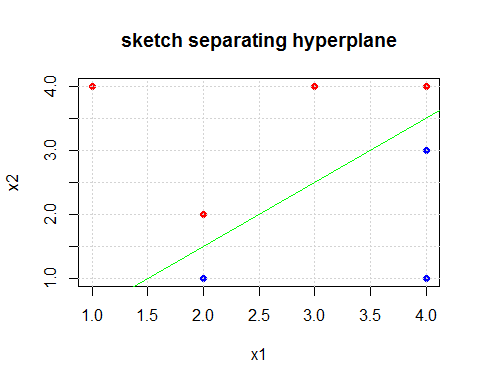


1. The hyperplane with maximum margin is one that separates the points with the space. Visually can see it is between the two sets of vertical red/blue points.

The equation, using y=ax+b, would be x2=x1-0.5. Rearranging this gives 0.5-x1+x2=0. However there is the additional rule that for the function f(B0+B1X1+B2\*X2), the sum of the beta's B0^2+ B1^2+ B2^2=1. So a scaling factor is needed:S(0.5^2+1+1)=1 --> which S=4/9 --> so all betas need to be scaled by 2/3.

In other words, the maximal margin classifier hyperplane should be 1/3-2x1/3+2x2/3=0

plot(x1,x2,col=y,pch=19, main='sketch separating hyperplane')  
grid()  
abline(-0.5,1,col='green')

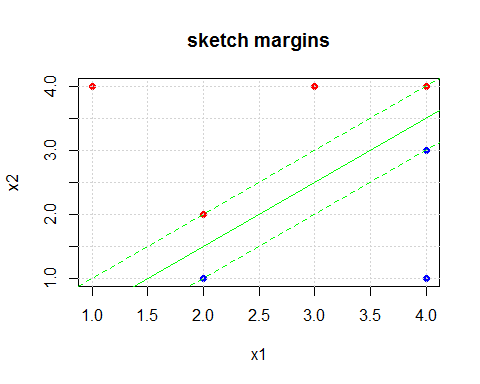


1. For the blue point (2,1) the equation (2/3)(0.5-x1+x2)=(2/3)(0.5-2+1)=(2/3)(-1/2)=(-1/3)<0 and for the red point (2,2), the equation gives (2/3)(0.5-2+2)=(1/3>0.

So the classification rule for the maximal margin classifier would be: Classify to Red if 1/3-2x1/3+2x2/3>0, and classify to Blue otherwise.

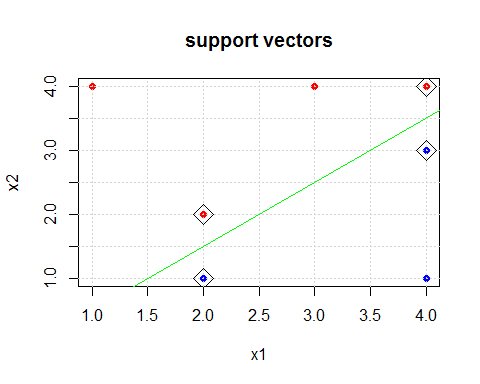
1. The margin is shown below in dashed green line: Margins are 0.5 units wide in each of the X1 direction and 0.5 units wide in each of the X2 direction.

plot(x1,x2,col=y,pch=19, main='sketch margins')  
grid()  
abline(-0.5,1,col='green')  
abline(-1,1,col='green',lty=2)  
abline(0,1,col='green',lty=2)



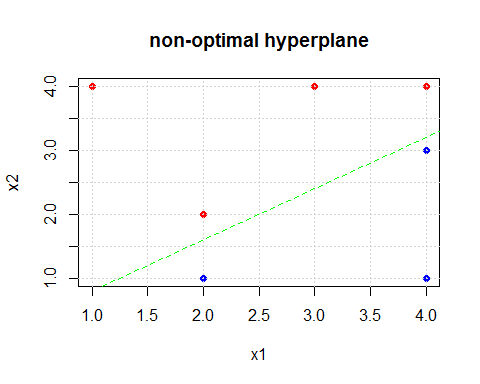
1. The support vectors are in diamonds in the plot:

library(e1071)  
dat<-data.frame(X1=x1,X2=x2,y=as.factor(y))  
svmfit<-svm(as.factor(y)~.,data=dat,scale=TRUE,cost=100,kernel='linear')  
plot(x1,x2,col=y,pch=19, main='support vectors')  
grid()  
points(dat[svmfit$index,],pch=5,cex=2)  
abline(-0.5,1,col='green')



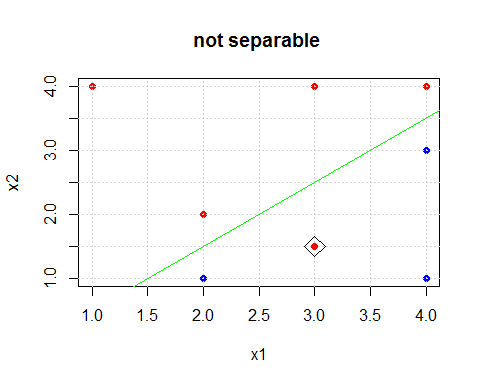
1. The seventh observation (4,1) is not a support vector so slight movements will not affect the maximal margin hyperplane.
2. A non optimal hyperplane is shown below:

plot(x1,x2,col=y,pch=19, main='non-optimal hyperplane')  
grid()  
abline(0,0.8,col='green',lty=2)



1. Additional observation such that it is not separable by hyperplane:

plot(x1,x2,col=y,pch=19, main='not separable ')  
grid()  
abline(-0.5,1,col='green')  
points(3,1.5,col='red',pch=19)  
points(3,1.5,pch=5,cex=2)



## Problem #3

1. Generate simulated data set with 20 observations in 3 classes and 50 variables. Use the same technique showed in the videos for k means

set.seed(2016)  
x=matrix(rnorm(3000),60,50)  
xmean=matrix(rnorm(150,sd=0.8),3,50)   
which=sample(1:3,60,replace=TRUE)  
x=x+xmean[which,]

1. Perform PCA and plot the first two principle component score vectors

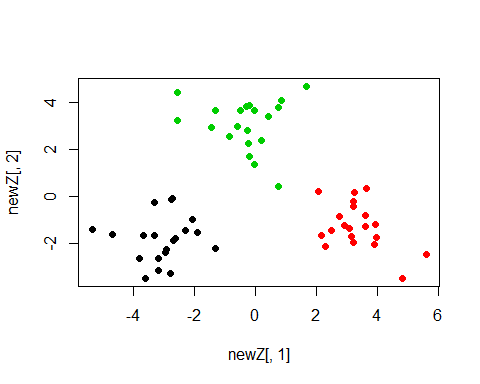
pca.p3=prcomp(x,scale=TRUE)  
names(pca.p3)

## [1] "sdev" "rotation" "center" "scale" "x"

#indeed deviation most in first two directions  
pca.p3$sdev

## [1] 2.8030721 2.4134517 1.6167619 1.5244812 1.4439299 1.3885805 1.3572352  
## [8] 1.3389972 1.2973757 1.2878625 1.2063245 1.1580062 1.1330475 1.0859169  
## [15] 1.0779180 1.0354186 1.0119037 0.9900842 0.9675981 0.9308868 0.8968074  
## [22] 0.8734349 0.8596657 0.7974815 0.7752768 0.7423121 0.7347774 0.6890229  
## [29] 0.6564992 0.6337629 0.6272555 0.5929882 0.5489029 0.5290253 0.5128413  
## [36] 0.5026460 0.4727421 0.4490142 0.4229210 0.3584712 0.3272142 0.3112547  
## [43] 0.2875585 0.2751544 0.2486320 0.2305399 0.1962775 0.1745152 0.1263476  
## [50] 0.1087742

loadings=pca.p3$rotation[,1:2]  
newZ=predict(pca.p3)  
plot(newZ[,1],newZ[,2],col=which,pch=19)



1. Perform k means with K=3. Assuming we know the number of cluster there are (K=3), it is able to separate the clusters well without any misclassifications.

set.seed(306)  
km.p3=kmeans(x,3,nstart=15)  
table(km.p3$cluster,which)

## which  
## 1 2 3  
## 1 20 0 0  
## 2 0 0 20  
## 3 0 20 0

1. Perform k means with K=2. Now it puts two clusters in one group, compared to the original 3 clusters.

set.seed(306)  
km.p3.k2=kmeans(x,2,nstart=15)  
table(km.p3.k2$cluster,which)

## which  
## 1 2 3  
## 1 20 0 20  
## 2 0 20 0

1. Perform k means with K=4. Now it split one of the 3 original clusters into two clusters.

set.seed(306)  
km.p3.k4=kmeans(x,4,nstart=15)  
table(km.p3.k4$cluster,which)

## which  
## 1 2 3  
## 1 20 0 0  
## 2 0 6 0  
## 3 0 0 20  
## 4 0 14 0

1. Perform k means with K=3 on the first two principal component score vectors. It actually is still able to cluster them correctly!

xwithZ<-newZ[,1:2]  
set.seed(306)  
km.p3.withZ<-kmeans(xwithZ,3,nstart=15)  
table(km.p3.withZ$cluster,which)

## which  
## 1 2 3  
## 1 20 0 0  
## 2 0 0 20  
## 3 0 20 0

1. Scaling k means. In this instance, they give identical results. This is useful if the components used have very different units and variances, but in this case they came from the same rnorm pool and does not have much of an effect.

set.seed(306)  
km.p3.scale<-kmeans(scale(x,center = FALSE, scale = apply(x, 2, sd)),3,nstart=15)  
table(km.p3.scale$cluster,which)

## which  
## 1 2 3  
## 1 20 0 0  
## 2 0 0 20  
## 3 0 20 0

## 

## Problem #4

1. Plot test MSE as a function of trees for bagging and random forest

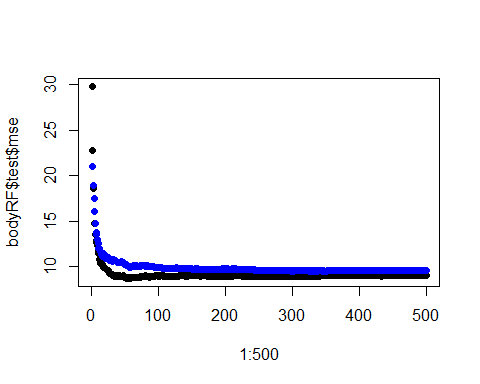
library(randomForest)

## Warning: package 'randomForest' was built under R version 3.2.3

## randomForest 4.6-12

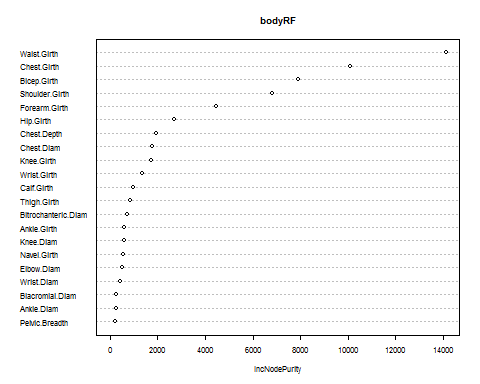
## Type rfNews() to see new features/changes/bug fixes.

load("C:/Users/licheong/Documents/Classes\_ISLR/body.RData")  
test<-sample(1:nrow(X),200)  
train<-(1:nrow(X))[-test]  
bodyDF<-data.frame(Weight=Y$Weight,X)  
bodyRF<-randomForest(Weight~.,data=bodyDF,subset=train,ntree=500,xtest=X[-train,],ytest=Y$Weight[-train])  
bodyBag<-randomForest(Weight~.,data=bodyDF,subset=train,ntree=500,xtest=X[-train,],ytest=Y$Weight[-train],mtry=21)  
plot(1:500, bodyRF$test$mse,pch=19)  
points(1:500,bodyBag$test$mse,pch=19,col='blue')

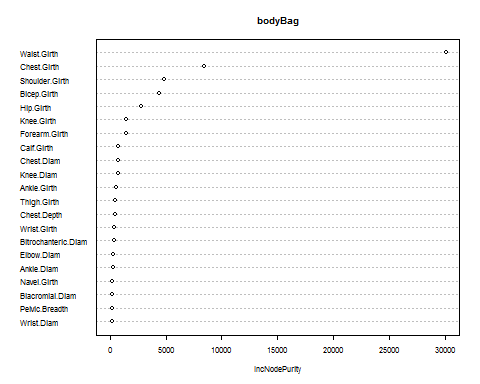


1. Variable importance between randomforest and bagging. Both found Waist.Girth as the most important variable, followed by Chest.Girth. They then disagree on the third most important variable, with randomForest determining that as Bicep.Girth while bagging determined Shoulder.Girth as third most important variable.

varImpPlot(bodyRF,cex=0.5)



varImpPlot(bodyBag,cex=0.5)



1. Compare test errors: From previous homework, the test error(PCR)=9.999427, test error (PLS)=9.218095,test error(FSS)=8.987679.  
   In comparison, the randomForest was worse than FSS but better than PCR and PLS.

bodyRF$test$mse[500]

## [1] 9.054664

1. No it would not be useful to include more test trees as the test error flattens out from about 50 trees onwards